AMENDMENTS TO THE CLAIMS

Claims 2, 9, 13, 18-30, 32-34, 36 and 37 are being amended, claims 14, 31, 38 and 39 are being canceled, and new claims 40-42 are being added. All pending claims are reproduced below, including those that remain unchanged.

1. (Original) A method for extending an existing programming language, comprising the steps of:

selecting an existing programming language; and,

extending an existing programming language by adding at least one language construct defined by a second language.

2. (Currently Amended) A method according to claim 1, wherein:

said existing programming language is ~~Java~~ an object oriented language.

3. (Original) A method according to claim 1, wherein:

said second language is XML.

4. (Original) A method according to claim 1, wherein:

said language construct is a parallelism construct representing parallel branch of program execution.

5. (Original) A method according to claim 4, wherein:

said parallelism construct further comprises plurality of branch constructs defined by said second language, wherein said branch constructs represent parallel branches of program execution comprising of at least one software activity.

6. (Original) A method according to claim 4, wherein:

said parallelism construct is further nested within a similar parallelism construct.

7. (Original) A method according to claim 1, wherein:

said language construct is a transaction construct representing transaction block of at least one software activity.

8. (Original) A method according to claim 7, wherein:

said transaction construct further specifies the number of retry attempts to perform the software activities inside said transaction block.

9. (Currently Amended) A method according to claim 7, wherein:

said transaction construct is further enclosed within a saga construct comprising [[of]] a compensation construct with at least one compensating software activity to undo work associated with the transaction block if the transaction block is aborted, ~~where in~~ wherein the saga construct represents a long running transaction.

10. (Original) A method according to claim 9, wherein:

said saga construct further comprises of plurality of transaction blocks.

11. (Original) A method according to claim 1, wherein:

said language construct is an exception handlers construct representing an execution mechanism comprising of exception handler construct defined by said second language, which represents exception not caught by the existing programming language handler methods.

12. (Original) A method according to claim 1, wherein:

said language construct is an action construct representing an activity that allows a first software component written using the extended existing programming language to call an operation on a second software component written using said existing programming language.

13. (Currently Amended) A method according to claim 12, wherein:

said action construct allows said first software component to call a piece of ~~Java~~ object oriented programming language code.

14. (Canceled)

15. (Original) A method according to claim 1, wherein:

said language construct is a multiple receive construct that allows a software component written using the extended existing programming language to wait on multiple input events received.

16. (Original) A method according to claim 15, wherein:

said multiple receive construct further allows said software component proceed on a particular branch of program execution, based on the input event that occurred first within the said multiple input events.

17. (Original) A method according to claim 1, wherein:

said language construct is a looping construct with ordering of messages received, representing looping functionality, wherein the ordering allows said messages to be received in an order.

18. (Currently Amended) A computer system ~~for~~ capable of extending an existing programming language, comprising:

an existing programming language stored in the computer system; and,

means for extending an existing programming language by adding at least one language construct defined by a second language.

19. (Currently Amended) A computer system according to claim 18, wherein:

said existing programming language is ~~Java~~ an object oriented programming language.

20. (Currently Amended) A computer system according to claim 18, wherein:

said second language is XML.

21. (Currently Amended) A computer system according to claim 18, wherein:

said language construct is a parallelism construct representing parallel branch of program execution.

22. (Currently Amended) A computer system according to claim 21, wherein:

said parallelism construct further comprises plurality of branch constructs defined by said second language, wherein said branch constructs represent parallel branches of program execution comprising of at least one software activity.

23. (Currently Amended) A computer system according to claim 21, wherein:

said parallelism construct is further nested within a similar parallelism construct.

24. (Currently Amended) A computer system according to claim 18, wherein:

said language construct is a transaction construct representing transaction block of at least one software activity.

25. (Currently Amended) A computer system according to claim 24, wherein:

said transaction construct further specifies the number of retry attempts to perform the software activities inside said transaction block.

26. (Currently Amended) A computer system according to claim 24, wherein:

said transaction construct is further enclosed within a saga construct comprising of compensation construct with at least one compensating software activity to undo work associated with the transaction block if the transaction block is aborted, where in the saga construct represents a long running transaction.

27. (Currently Amended) A computer system according to claim 26, wherein:

said saga construct further comprises of plurality of transaction blocks.

28. (Currently Amended) A computer system according to claim 18, wherein:

said language construct is an exception handlers construct representing an execution mechanism comprising of exception handler construct defined by said second language, which represents exception not caught by the existing programming language handler methods.

29. (Currently Amended) A computer system according to claim 18, wherein:

said language construct is an action construct representing an activity that allows a first software component written using the extended existing programming language to call an operation on a second software component written using said existing programming language.

30. (Currently Amended) A computer system according to claim 29, wherein:

said action construct allows said first software component to call a piece of ~~Java~~ object oriented programming language code.

31. (Canceled)

32. (Currently Amended) A computer system according to claim 18, wherein:

said language construct is a multiple receive construct that allows a software component written using the extended existing programming language to wait on multiple input events received.

33. (Currently Amended) A computer system according to claim 32, wherein:

said multiple receive construct further allows said software component proceed on a particular branch of program execution, based on the input event that occurred first within the said multiple input events.

34. (Currently Amended) A computer system according to claim 18, wherein:

said language construct is a looping construct with ordering of messages received, representing looping functionality, wherein the ordering allows said messages to be received in an order.

35. (Original) A computer system comprising:

a processor;

object code executed by said processor, said object code configured to:

extend an existing programming language by adding a language construct defined by a second language.

36. (Currently Amended) A method for extending ~~Java~~ object oriented programming language, comprising the steps of:

selecting ~~Java~~ objected oriented programming language; and,

extending ~~Java~~ the object oriented programming language by adding at least one language construct defined by XML.

37. (Currently Amended) A storage medium including code stored thereon and executable by a computer system for extending ~~Java~~ an objected oriented programming language, comprising:

~~a Java~~ an object oriented programming language; and,

means for extending ~~Java~~ the object oriented programming language by adding at least one language construct defined by XML.

38.-39. (Canceled)

40. (New) A system for utilizing a workflow language, comprising:

a computer including a processing device operating thereon;

a source file stored on a computer readable medium, wherein the source file includes a workflow definition created using a workflow language, wherein said workflow language comprises an object oriented programming language extended with a plurality of workflow constructs defined by XML, including constructs for defining loop processing that performs a plurality of activities so long as a condition attribute is true; and

means for creating a workflow program according to said workflow definition, including

means for the computer to read the source file and process the plurality of activities,

means for determining whether the condition attribute is true,

means for performing the plurality of activities when the condition attribute is true, and

means for terminating the loop processing when the condition attribute is not true.

41. (New) The system of claim 40, wherein the means for determining whether the condition attribute is true, determines whether the condition attribute is true before the plurality of activities are performed.

42. (New) The system of claim 40, wherein the means for determining whether the condition attribute is true, determines whether the condition attribute is true after the plurality of activities are performed.